Non-intrusive model reduction of shift-equivariant systems via data-driven projection and reduced dynamics

Yu Shuai

September 9, 2025

2025-09-09

Non-intrusive model reduction of shift-equivariant systems via data-driven projection and reduced dynamics

Non-intrusive model reduction of shift-equivariant systems via data-driven projection and reduced dynamics

Yu Shuai

September 9, 2025

Contents

1 Review: Symmetry-reduced operator inference (SR-OpInf)

2 Background

3 Method

Contents

1 Review: Symmetry-reduced operator inference (SR-OpInf)
2 Background
3 Method

Contents

1 Review: Symmetry-reduced operator inference (SR-OpInf)
- SR-OpInf for model reduction

2 Background

3 Method

Contents

1 Review: Symmetry-reduced operator inference (SR-OpInf)
- SR-OpInf for model reduction

2 Background

3 Method

SR-OpInf for model reduction

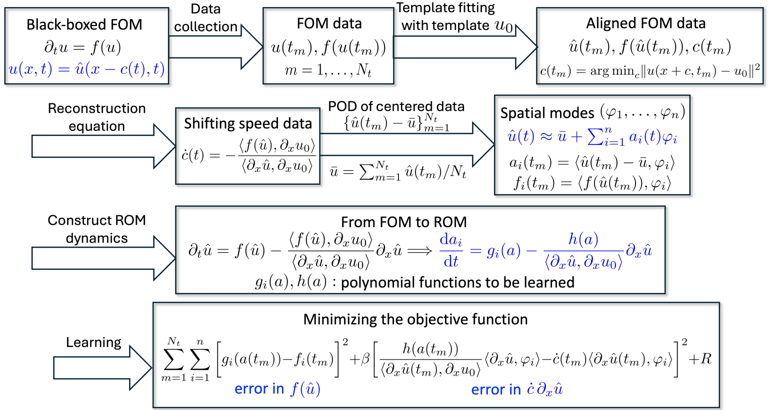
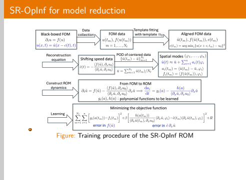


Figure: Training procedure of the SR-OpInf ROM



Here I use this diagram to show the algorithm of symmetry-reduced operator inference. We start from a black-boxed full-order model partial u partial t = f(u). The system is shift equivariant, means that you can move your solution, and it has traveling solutions. We want to predict those traveling solutions uhat using a small number of modes. That's our goal. To do this, we train a model. We first collect some data, and then perform template fitting to align those traveling solutions against a template to remove their drifting. From this fitting, we can get the aligned solutions, their RHSs and the shifting amount. Based on this, we utilize the reconstruction equation to compute their traveling speeds. Once we collect these data, we now try to approximate the frozen solution profiles using POD modes, and assume a symmetry-reduced ROM based on the full-order dynamics of fitted solution uhat. Finally, we learn the functions.

SR-OpInf for model reduction

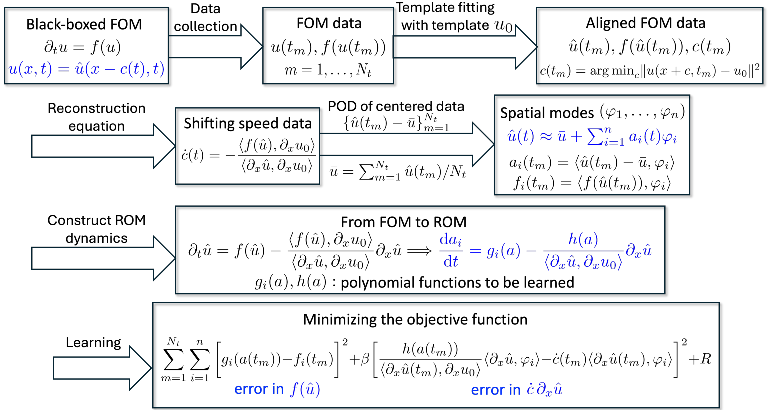
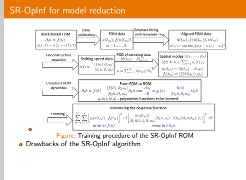
- 

  Figure: Training procedure of the SR-OpInf ROM
- Drawbacks of the SR-OpInf algorithm



Here I use this diagram to show the algorithm of symmetry-reduced operator inference. We start from a black-boxed full-order model partial u partial t = f(u). The system is shift equivariant, means that you can move your solution, and it has traveling solutions. We want to predict those traveling solutions uhat using a small number of modes. That's our goal. To do this, we train a model. We first collect some data, and then perform template fitting to align those traveling solutions against a template to remove their drifting. From this fitting, we can get the aligned solutions, their RHSs and the shifting amount. Based on this, we utilize the reconstruction equation to compute their traveling speeds. Once we collect these data, we now try to approximate the frozen solution profiles using POD modes, and assume a symmetry-reduced ROM based on the full-order dynamics of fitted solution uhat. Finally, we learn the functions.

SR-OpInf for model reduction

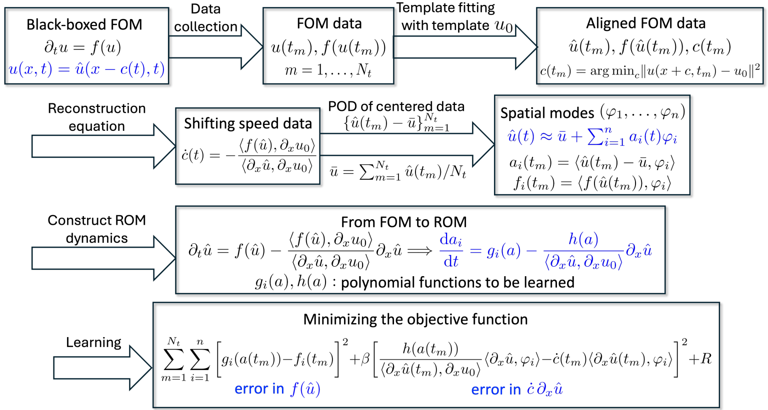
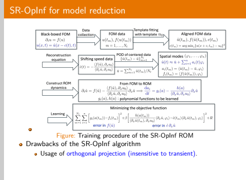


Figure: Training procedure of the SR-OpInf ROM

- Drawbacks of the SR-OpInf algorithm
  - Usage of orthogonal projection (insensitive to transient).



Here I use this diagram to show the algorithm of symmetry-reduced operator inference. We start from a black-boxed full-order model partial u partial t = f(u). The system is shift equivariant, means that you can move your solution, and it has traveling solutions. We want to predict those traveling solutions uhat using a small number of modes. That's our goal. To do this, we train a model. We first collect some data, and then perform template fitting to align those traveling solutions against a template to remove their drifting. From this fitting, we can get the aligned solutions, their RHSs and the shifting amount. Based on this, we utilize the reconstruction equation to compute their traveling speeds. Once we collect these data, we now try to approximate the frozen solution profiles using POD modes, and assume a symmetry-reduced ROM based on the full-order dynamics of fitted solution uhat. Finally, we learn the functions.

SR-OpInf for model reduction

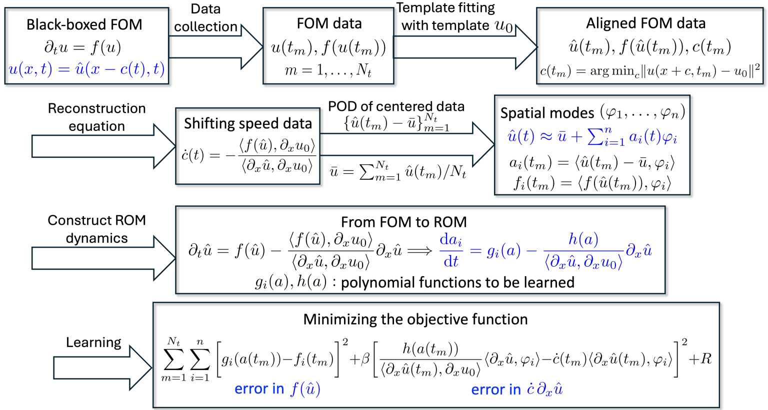
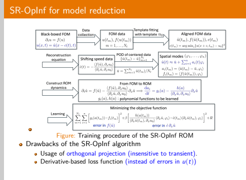


Figure: Training procedure of the SR-OpInf ROM

- Drawbacks of the SR-OpInf algorithm
  - Usage of orthogonal projection (insensitive to transient).
  - Derivative-based loss function (instead of errors in $u(t)$)



Here I use this diagram to show the algorithm of symmetry-reduced operator inference. We start from a black-boxed full-order model partial u partial t = f(u). The system is shift equivariant, means that you can move your solution, and it has traveling solutions. We want to predict those traveling solutions uhat using a small number of modes. That's our goal. To do this, we train a model. We first collect some data, and then perform template fitting to align those traveling solutions against a template to remove their drifting. From this fitting, we can get the aligned solutions, their RHSs and the shifting amount. Based on this, we utilize the reconstruction equation to compute their traveling speeds. Once we collect these data, we now try to approximate the frozen solution profiles using POD modes, and assume a symmetry-reduced ROM based on the full-order dynamics of fitted solution uhat. Finally, we learn the functions.

SR-OpInf for model reduction

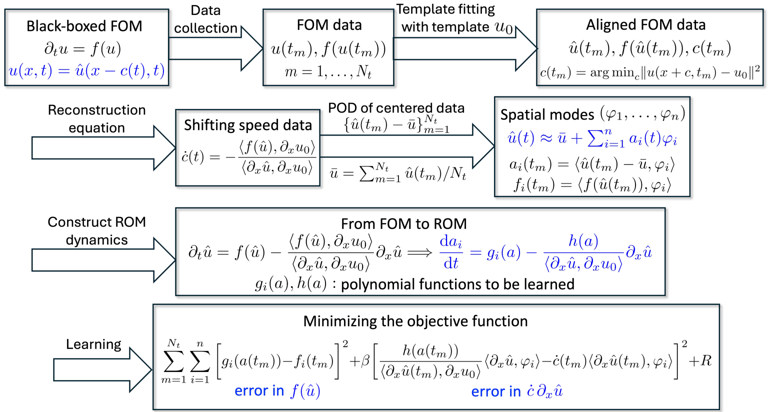
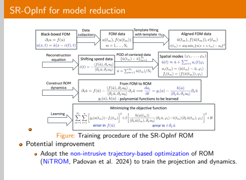


Figure: Training procedure of the SR-OpInf ROM

- Potential improvement
  - Adopt the non-intrusive trajectory-based optimization of ROM (NiTROM, Padovan et al. 2024) to train the projection and dynamics.

Non-intrusive model reduction of shift-equivariant systems via data-driven projection and reduced dynamics
└Review: Symmetry-reduced operator inference (SR-OpInf)



Contents

1 Review: Symmetry-reduced operator inference (SR-OpInf)

2 Background
- Introduction to NiTROM

3 Method

Contents

Review: Symmetry-reduced operator inference (SR-OpInf)

Background

Introduction to NiTROM

Method

NiTROM: non-intrusive trajectory-based ROM

- Consider a general nonlinear FOM (no need to be shift-equivariant):

$$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u), u \in \mathbb{R}^n, u(0) = u_0 \tag{1}$$

NiTROM: non-intrusive trajectory-based ROM
- Consider a general nonlinear FOM (no need to be shift-equivariant):
$$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u), u \in \mathbb{R}^n, u(0) = u_0 \tag{1}$$

NiTROM: non-intrusive trajectory-based ROM

- Consider a general nonlinear FOM (no need to be shift-equivariant):

$$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u), u \in \mathbb{R}^n, u(0) = u_0 \tag{1}$$

- Define an oblique projection with two matrices $\Phi, \Psi \in \mathbb{R}^{n\times r}$

$$a = \Psi^\top u \in \mathbb{R}^r \tag{2a}$$

$$u \approx \Phi(\Psi^\top \Phi)^{-1} a \tag{2b}$$

NiTROM: non-intrusive trajectory-based ROM
Consider a general nonlinear FOM (no need to be shift-equivariant):
Define an oblique projection with two matrices $\Phi, \Psi \in \mathbb{R}^{n\times r}$

NiTROM: non-intrusive trajectory-based ROM

- Consider a general nonlinear FOM (no need to be shift-equivariant):

$$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u), u \in \mathbb{R}^n, u(0) = u_0 \tag{1}$$

- Define an oblique projection with two matrices $\Phi, \Psi \in \mathbb{R}^{n \times r}$

$$a = \Psi^\top u \in \mathbb{R}^r \tag{2a}$$

$$u \approx \Phi(\Psi^\top \Phi)^{-1} a \tag{2b}$$

  - If $\Phi = \Psi$, then the projection is orthogonal (e.g., POD basis matrix)

NiTROM: non-intrusive trajectory-based ROM
Consider a general nonlinear FOM (no need to be shift-equivariant):
$$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u), u \in \mathbb{R}^n, u(0) = u_0 \tag{1}$$
Define an oblique projection with two matrices $\Phi, \Psi \in \mathbb{R}^{n \times r}$
$$a = \Psi^\top u \in \mathbb{R}^r \tag{2a}$$
$$u \approx \Phi(\Psi^\top \Phi)^{-1} a \tag{2b}$$
If $\Phi = \Psi$, then the projection is orthogonal (e.g., POD basis matrix)

NiTROM: non-intrusive trajectory-based ROM

- Consider a general nonlinear FOM (no need to be shift-equivariant):

$$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u), u \in \mathbb{R}^n, u(0) = u_0 \tag{1}$$

- Define an oblique projection with two matrices $\Phi, \Psi \in \mathbb{R}^{n \times r}$

$$a = \Psi^\top u \in \mathbb{R}^r \tag{2a}$$

$$u \approx \Phi(\Psi^\top \Phi)^{-1} a \tag{2b}$$

  - If $\Phi = \Psi$, then the projection is orthogonal (e.g., POD basis matrix)
- The non-intrusive ROM then takes the form of

$$\frac{\mathrm{d}a}{\mathrm{d}t} = g(a), a(0) = \Psi^\top u(0) \tag{3a}$$

$$u \approx u_r = \Phi(\Psi^\top \Phi)^{-1} a \tag{3b}$$

NiTROM: non-intrusive trajectory-based ROM
Consider a general nonlinear FOM (no need to be shift-equivariant):
$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u), u \in \mathbb{R}^n, u(0) = u_0$ (1)
Define an oblique projection with two matrices $\Phi, \Psi \in \mathbb{R}^{n \times r}$
$a = \Psi^\top u \in \mathbb{R}^r$ (2a)
$u \approx \Phi(\Psi^\top \Phi)^{-1} a$ (2b)
If $\Phi = \Psi$, then the projection is orthogonal (e.g., POD basis matrix)
The non-intrusive ROM then takes the form of
$\frac{\mathrm{d}a}{\mathrm{d}t} = g(a), a(0) = \Psi^\top u(0)$ (3a)
$u \approx u_r = \Phi(\Psi^\top \Phi)^{-1} a$ (3b)

NiTROM: non-intrusive trajectory-based ROM

- Consider a general nonlinear FOM (no need to be shift-equivariant):

$$\frac{\mathrm{d}u}{\mathrm{d}t} = f(u), u \in \mathbb{R}^n, u(0) = u_0 \tag{1}$$

- Define an oblique projection with two matrices $\Phi, \Psi \in \mathbb{R}^{n \times r}$

$$a = \Psi^\top u \in \mathbb{R}^r \tag{2a}$$

$$u \approx \Phi(\Psi^\top \Phi)^{-1} a \tag{2b}$$

  - If $\Phi = \Psi$, then the projection is orthogonal (e.g., POD basis matrix)
- The non-intrusive ROM then takes the form of

$$\frac{\mathrm{d}a}{\mathrm{d}t} = g(a), a(0) = \Psi^\top u(0) \tag{3a}$$

$$u \approx u_r = \Phi(\Psi^\top \Phi)^{-1} a \tag{3b}$$

  - $g(a) = Aa + B : (aa^\top) = A_{ij} a_j + B_{ijk} a_j a_k, A \in \mathbb{R}^{r \times r}, B \in \mathbb{R}^{r \times r \times r}$.

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.
- However, two difficulties prevent us from updating variables in standard Euclidean space of matrices.

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.
- However, two difficulties prevent us from updating variables in standard Euclidean space of matrices.

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.
- However, two difficulties prevent us from updating variables in standard Euclidean space of matrices.
  - Φ and ΦQ gives the same decoder if Q is any invertible r-by-r matrix.

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a = (\Phi Q)(\Psi^\top (\Phi Q))^{-1} a \tag{4}$$

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.
- However, two difficulties prevent us from updating variables in standard Euclidean space of matrices.
  - Φ and ΦQ gives the same decoder if Q is any invertible r-by-r matrix.

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a = (\Phi Q)(\Psi^\top (\Phi Q))^{-1} a \tag{4}$$

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.
- However, two difficulties prevent us from updating variables in standard Euclidean space of matrices.
  - Φ and ΦQ gives the same decoder if Q is any invertible r-by-r matrix.

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a = (\Phi Q)(\Psi^\top(\Phi Q))^{-1} a \tag{4}$$

- Thus, the actual variable to be trained is the r-dim subspace V spanned by Φ.

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.
- However, two difficulties prevent us from updating variables in standard Euclidean space of matrices.
  - Φ and ΦQ gives the same decoder if Q is any invertible r-by-r matrix.

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a = (\Phi Q)(\Psi^\top(\Phi Q))^{-1} a \tag{4}$$

- Thus, the actual variable to be trained is the r-dim subspace V spanned by Φ.

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.
- However, two difficulties prevent us from updating variables in standard Euclidean space of matrices.
  - Φ and ΦQ gives the same decoder if Q is any invertible r-by-r matrix.

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a = (\Phi Q)(\Psi^\top (\Phi Q))^{-1} a \tag{4}$$

    - Thus, the actual variable to be trained is the r-dim subspace V spanned by Φ.
  - Both Φ and Ψ should have full column rank to ensure $(\Psi^\top \Phi)^{-1}$ exists.

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.
- However, two difficulties prevent us from updating variables in standard Euclidean space of matrices.
  - Φ and ΦQ gives the same decoder if Q is any invertible r-by-r matrix.

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a = (\Phi Q)(\Psi^\top (\Phi Q))^{-1} a \tag{4}$$

    - Thus, the actual variable to be trained is the r-dim subspace V spanned by Φ.
  - Both Φ and Ψ should have full column rank to ensure $(\Psi^\top \Phi)^{-1}$ exists.

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.
- However, two difficulties prevent us from updating variables in standard Euclidean space of matrices.
  - Φ and ΦQ gives the same decoder if Q is any invertible r-by-r matrix.

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a = (\Phi Q)(\Psi^\top (\Phi Q))^{-1} a \tag{4}$$

    - Thus, the actual variable to be trained is the r-dim subspace V spanned by Φ.
  - Both Φ and Ψ should have full column rank to ensure $(\Psi^\top \Phi)^{-1}$ exists.
    - A natural way is to constrain Ψ to have orthonormal columns.

Trajectory-based optimization of non-intrusive ROM

- At present, the variables to be optimized are bases Φ, Ψ and coefficients A, B.
- However, two difficulties prevent us from updating variables in standard Euclidean space of matrices.
  - Φ and ΦQ gives the same decoder if Q is any invertible r-by-r matrix.

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a = (\Phi Q)(\Psi^\top (\Phi Q))^{-1} a \tag{4}$$

    - Thus, the actual variable to be trained is the r-dim subspace V spanned by Φ.
  - Both Φ and Ψ should have full column rank to ensure $(\Psi^\top \Phi)^{-1}$ exists.
    - A natural way is to constrain Ψ to have orthonormal columns.

Trajectory-based optimization of non-intrusive ROM

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B : (aa^\top), a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top \Psi = I_r. \tag{5d}$$

Trajectory-based optimization of non-intrusive ROM

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B : (aa^\top), a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top \Psi = I_r. \tag{5d}$$

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B : (aa^\top),\, a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top\Phi)^{-1}a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top\Psi = I_r. \tag{5d}$$

- To solve this problem, we need to:

Trajectory-based optimization of non-intrusive ROM

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B : (aa^\top),\, a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top\Phi)^{-1}a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top\Psi = I_r. \tag{5d}$$

- To solve this problem, we need to:

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B : (aa^\top), a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top \Psi = I_r. \tag{5d}$$

- To solve this problem, we need to:
  - Identify the domain (i.e. manifold) for each variable.

Trajectory-based optimization of non-intrusive ROM
We now state the optimization problem as follows:
To solve this problem, we need to:
Identify the domain (i.e. manifold) for each variable.

Trajectory-based optimization of non-intrusive ROM

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B : (aa^\top), a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top \Psi = I_r. \tag{5d}$$

- To solve this problem, we need to:
  - Identify the domain (i.e. manifold) for each variable.
  - Compute the gradient of the objective on the manifold.

Trajectory-based optimization of non-intrusive ROM

We now state the optimization problem as follows:

To solve this problem, we need to:
Identify the domain (i.e. manifold) for each variable.
Compute the gradient of the objective on the manifold.

Trajectory-based optimization of non-intrusive ROM

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B : (aa^\top),\ a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top \Phi)^{-1} a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top \Psi = I_r. \tag{5d}$$

- To solve this problem, we need to:
  - Identify the domain (i.e. manifold) for each variable.
  - Compute the gradient of the objective on the manifold.
  - Update the variables while keeping them on their manifolds.

Trajectory-based optimization of non-intrusive ROM

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B : (aa^\top),\ a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top\Phi)^{-1}a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top\Psi = I_r. \tag{5d}$$

- To solve this problem, we need to:
  - Identify the domain (i.e. manifold) for each variable.
  - Compute the gradient of the objective on the manifold.
  - Update the variables while keeping them on their manifolds.
- Toolbox: Pymanopt (Townsend et al. 2016)

2025-09-09

Non-intrusive model reduction of shift-equivariant systems via data-driven projection and reduced dynamics
└Background
└Introduction to NiTROM

Trajectory-based optimization of non-intrusive ROM
We now state the optimization problem as follows:
To solve this problem, we need to:
Identify the domain (i.e. manifold) for each variable.
Compute the gradient of the objective on the manifold.
Update the variables while keeping them on their manifolds.
Toolbox: Pymanopt (Townsend et al. 2016)

Trajectory-based optimization of non-intrusive ROM

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B:(aa^\top),\ a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top\Phi)^{-1}a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top\Psi = I_r. \tag{5d}$$

- To solve this problem, we need to:
  - Identify the domain (i.e. manifold) for each variable.
  - Compute the gradient of the objective on the manifold.
  - Update the variables while keeping them on their manifolds.
- Toolbox: Pymanopt (Townsend et al. 2016)
  - Automatically handles all these issues.

Trajectory-based optimization of non-intrusive ROM
We now state the optimization problem as follows:
To solve this problem, we need to:
Identify the domain (i.e. manifold) for each variable.
Compute the gradient of the objective on the manifold.
Update the variables while keeping them on their manifolds.
Toolbox: Pymanopt (Townsend et al. 2016)
Automatically handles all these issues.

Trajectory-based optimization of non-intrusive ROM

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B:(aa^\top),\ a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top\Phi)^{-1}a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top\Psi = I_r. \tag{5d}$$

- To solve this problem, we need to:
  - Identify the domain (i.e. manifold) for each variable.
  - Compute the gradient of the objective on the manifold.
  - Update the variables while keeping them on their manifolds.
- Toolbox: Pymanopt (Townsend et al. 2016)
  - Automatically handles all these issues.
  - Allows us to update V by updating Φ.

Trajectory-based optimization of non-intrusive ROM

- We now state the optimization problem as follows:
- To solve this problem, we need to:
  - Identify the domain (i.e. manifold) for each variable.
  - Compute the gradient of the objective on the manifold.
  - Update the variables while keeping them on their manifolds.
- Toolbox: Pymanopt (Townsend et al. 2016)
  - Automatically handles all these issues.
  - Allows us to update V by updating Φ.

- We now state the optimization problem as follows:

$$\min_{(V,\Psi,A,B)} J = \sum_{m=0}^{N_t-1} \|u(t_i) - u_r(t_i)\|^2 \quad \text{(trajectory-based error)} \tag{5a}$$

$$s.t. \quad \frac{\mathrm{d}a}{\mathrm{d}t} = g(a) = Aa + B : (aa^\top), a(t_0) = \Psi^\top u(t_0) \tag{5b}$$

$$u_r = \Phi(\Psi^\top\Phi)^{-1}a \tag{5c}$$

$$V = \mathrm{Range}(\Phi), \mathrm{rank}(\Phi) = r, \Psi^\top\Psi = I_r. \tag{5d}$$

- To solve this problem, we need to:
  - Identify the domain (i.e. manifold) for each variable.
  - Compute the gradient of the objective on the manifold.
  - Update the variables while keeping them on their manifolds.
- Toolbox: Pymanopt (Townsend et al. 2016)
  - Automatically handles all these issues.
  - Allows us to update V by updating Φ.
  - Only needs user-input standard derivatives of J w.r.t. (Φ, Ψ, A, B).

2025-09-09

Non-intrusive model reduction of shift-equivariant systems via data-driven projection and reduced dynamics
└Background
└Introduction to NiTROM

Trajectory-based optimization of non-intrusive ROM
We now state the optimization problem as follows:
To solve this problem, we need to:
Identify the domain (i.e. manifold) for each variable.
Compute the gradient of the objective on the manifold.
Update the variables while keeping them on their manifolds.
Toolbox: Pymanopt (Townsend et al. 2016)
Automatically handles all these issues.
Allows us to update V by updating Φ.
Only needs user-input standard derivatives of J w.r.t. (Φ, Ψ, A, B).

Contents

1 Review: Symmetry-reduced operator inference (SR-OpInf)

2 Background

3 Method
- A symmetry-reduced NiTROM (SR-NiTROM) for shift-equivariant systems

Contents

1 Review: Symmetry-reduced operator inference (SR-OpInf)

2 Background

3 Method
- A symmetry-reduced NiTROM (SR-NiTROM) for shift-equivariant systems

Optimization problem of SR-NiTROM

- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.

Optimization problem of SR-NiTROM
Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.

Optimization problem of SR-NiTROM

- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.

Optimization problem of SR-NiTROM
- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.

Optimization problem of SR-NiTROM

- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
- Objective function: trajectory-based errors of template-fitted profile and shifting amount

$$J(V, \Psi, A, B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2 \quad (6)$$

Optimization problem of SR-NiTROM

- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
- Objective function: trajectory-based errors of template-fitted profile and shifting amount

$$J(V, \Psi, A, B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2 \quad (6)$$

Optimization problem of SR-NiTROM

- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
- Objective function: trajectory-based errors of template-fitted profile and shifting amount

$$J(V, \Psi, A, B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2 \quad (6)$$

  - β: relative weight $\sim \|u\|^2/(dx)^2$

Optimization problem of SR-NiTROM
- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
- Objective function: trajectory-based errors of template-fitted profile and shifting amount
$$J(V, \Psi, A, B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2 \quad (6)$$
  - β: relative weight $\sim \|u\|^2/(dx)^2$

Optimization problem of SR-NiTROM

- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
- Objective function: trajectory-based errors of template-fitted profile and shifting amount

$$J(V,\Psi,A,B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2 \quad (6)$$

  - β: relative weight $\sim \|u\|^2/(dx)^2$
- Constraints:

Optimization problem of SR-NiTROM
Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
Objective function: trajectory-based errors of template-fitted profile and shifting amount
$J(V,\Psi,A,B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2$ (6)
β: relative weight $\sim \|u\|^2/(dx)^2$
Constraints:

Optimization problem of SR-NiTROM

- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
- Objective function: trajectory-based errors of template-fitted profile and shifting amount

$$J(V, \Psi, A, B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2 \quad (6)$$

  - β: relative weight $\sim \|u\|^2/(dx)^2$
- Constraints:
  - decoder & encoded initial value: $\widehat{u}_r = \Phi(\Psi^\top\Phi)^{-1}a$, $a(t_0) = \Psi^\top\widehat{u}(t_0)$

Optimization problem of SR-NiTROM

- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
- Objective function: trajectory-based errors of template-fitted profile and shifting amount

$$J(V, \Psi, A, B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2 \quad (6)$$

  - β: relative weight $\sim \|u\|^2/(dx)^2$
- Constraints:
  - decoder & encoded initial value: $\widehat{u}_r = \Phi(\Psi^\top\Phi)^{-1}a$, $a(t_0) = \Psi^\top\widehat{u}(t_0)$

Optimization problem of SR-NiTROM

- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
- Objective function: trajectory-based errors of template-fitted profile and shifting amount

$$J(V, \Psi, A, B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2 \quad (6)$$

  - β: relative weight $\sim \|u\|^2/(dx)^2$
- Constraints:
  - decoder & encoded initial value: $\widehat{u}_r = \Phi(\Psi^\top\Phi)^{-1}a$, $a(t_0) = \Psi^\top\widehat{u}(t_0)$
  - profile equation:

$$\frac{\mathrm{d}a}{\mathrm{d}t} = Aa + B : (aa^\top) + \frac{\mathrm{d}c_r}{\mathrm{d}t}(\Psi^\top\partial_x\widehat{u}_r) \quad (7)$$

2025-09-09

Non-intrusive model reduction of shift-equivariant systems via data-driven projection and reduced dynamics
└Method
  └A symmetry-reduced NiTROM (SR-NiTROM) for

Optimization problem of SR-NiTROM
Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
Objective function: trajectory-based errors of template-fitted profile and shifting amount
$J(V, \Psi, A, B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2$ (6)
β: relative weight $\sim \|u\|^2/(dx)^2$
Constraints:
decoder & encoded initial value: $\widehat{u}_r = \Phi(\Psi^\top\Phi)^{-1}a$, $a(t_0) = \Psi^\top\widehat{u}(t_0)$
profile equation:
$\frac{\mathrm{d}a}{\mathrm{d}t} = Aa + B : (aa^\top) + \frac{\mathrm{d}c_r}{\mathrm{d}t}(\Psi^\top\partial_x\widehat{u}_r)$ (7)

Optimization problem of SR-NiTROM

- Goal: Reduce a n-dim shift-equivariant FOM to a r-dim ROM.
- Data: trajectory snapshots $\{u(t_m)\}_{m=0}^{N_t-1}$ from a shift-equivariant system. A template u_0 for template fitting $u \to \widehat{u}$.
- Objective function: trajectory-based errors of template-fitted profile and shifting amount

$$J(V, \Psi, A, B) = \sum_{m=0}^{N_t-1} \|\widehat{u}(t_m) - \widehat{u}_r(t_m)\|^2 + \beta\Big(c(t_m) - c_r(t_m)\Big)^2 \quad (6)$$

  - β: relative weight $\sim \|u\|^2/(dx)^2$
- Constraints:
  - decoder & encoded initial value: $\widehat{u}_r = \Phi(\Psi^\top\Phi)^{-1}a$, $a(t_0) = \Psi^\top\widehat{u}(t_0)$
  - profile equation:

$$\frac{\mathrm{d}a}{\mathrm{d}t} = Aa + B : (aa^\top) + \frac{\mathrm{d}c_r}{\mathrm{d}t}(\Psi^\top\partial_x\widehat{u}_r) \quad (7)$$

  - velocity equation:

$$\frac{\mathrm{d}c_r}{\mathrm{d}t} = -\frac{p^\top a + a^\top Qa}{\langle\partial_x\widehat{u}, \partial_x u_0\rangle} \quad (8)$$

Optimization problem of SR-NiTROM

- Unconstrained optimization problem:

$$L(\Phi, \Psi, A, B, p, Q) = \sum_{m=0}^{N_t-1} L_m$$

$$\begin{aligned}
L_m = {} & \|\Phi(\Psi^\top \Phi)^{-1} a(t_m) - \widehat{u}(t_m)\|_2^2 + \beta\Big(c_r(t_m) - c(t_m)\Big)^2 \\
& + \int_{t_0}^{t_m} \lambda_m^\top(t)\left(\frac{\mathrm{d}a}{\mathrm{d}t} - Aa - B : (aa^\top) - \frac{\mathrm{d}c_r}{\mathrm{d}t}\Psi^\top \partial_x \Phi(\Psi^\top\Phi)^{-1} a\right)\mathrm{d}t \\
& + \int_{t_0}^{t_m} \mu_m(t)\left(\frac{\mathrm{d}c_r}{\mathrm{d}t} + \frac{p^\top a + a^\top Q a}{\langle \partial_x \Phi(\Psi^\top\Phi)^{-1} a, \partial_x u_0\rangle}\right)\mathrm{d}t \\
& + \lambda_m^\top(t_0)\Big(a(t_0) - \Psi^\top \widehat{u}(t_0)\Big)
\end{aligned} \tag{9}$$

Optimization problem of SR-NiTROM
Unconstrained optimization problem:

- Unconstrained optimization problem:

$$
\begin{aligned}
L(\Phi, \Psi, A, B, p, Q) &= \sum_{m=0}^{N_t-1} L_m \\
L_m &= \|\Phi(\Psi^\top\Phi)^{-1}a(t_m) - \widehat{u}(t_m)\|_2^2 + \beta\Big(c_r(t_m) - c(t_m)\Big)^2 \\
&+ \int_{t_0}^{t_m} \lambda_m^\top(t)\left(\frac{\mathrm{d}a}{\mathrm{d}t} - Aa - B:(aa^\top) - \frac{\mathrm{d}c_r}{\mathrm{d}t}\Psi^\top\partial_x\Phi(\Psi^\top\Phi)^{-1}a\right)\mathrm{d}t \\
&+ \int_{t_0}^{t_m} \mu_m(t)\left(\frac{\mathrm{d}c_r}{\mathrm{d}t} + \frac{p^\top a + a^\top Q a}{\langle \partial_x\Phi(\Psi^\top\Phi)^{-1}a, \partial_x u_0\rangle}\right)\mathrm{d}t \\
&+ \lambda_m^\top(t_0)\Big(a(t_0) - \Psi^\top\widehat{u}(t_0)\Big) \qquad (9)
\end{aligned}
$$

  - $\lambda_m(t) \in \mathbb{R}^n, \mu_m(t) \in \mathbb{R}$: Lagrangian multipliers.

Optimization problem of SR-NiTROM
Unconstrained optimization problem:
$L(\Phi, \Psi, A, B, p, Q) = \sum_{m=0}^{N_t-1} L_m$
$L_m = \|\Phi(\Psi^\top\Phi)^{-1}a(t_m) - \widehat{u}(t_m)\|_2^2 + \beta\Big(c_r(t_m) - c(t_m)\Big)^2 + \int_{t_0}^{t_m} \lambda_m^\top(t)\left(\frac{\mathrm{d}a}{\mathrm{d}t} - Aa - B:(aa^\top) - \frac{\mathrm{d}c_r}{\mathrm{d}t}\Psi^\top\partial_x\Phi(\Psi^\top\Phi)^{-1}a\right)\mathrm{d}t + \int_{t_0}^{t_m} \mu_m(t)\left(\frac{\mathrm{d}c_r}{\mathrm{d}t} + \frac{p^\top a + a^\top Q a}{\langle \partial_x\Phi(\Psi^\top\Phi)^{-1}a, \partial_x u_0\rangle}\right)\mathrm{d}t + \lambda_m^\top(t_0)\Big(a(t_0) - \Psi^\top\widehat{u}(t_0)\Big) \quad (9)$
$\lambda_m(t) \in \mathbb{R}^n, \mu_m(t) \in \mathbb{R}$: Lagrangian multipliers.

Optimization problem of SR-NiTROM

- Unconstrained optimization problem:

$$L(\Phi, \Psi, A, B, p, Q) = \sum_{m=0}^{N_t-1} L_m$$

$$\begin{aligned}
L_m &= \|\Phi(\Psi^\top\Phi)^{-1}a(t_m) - \widehat{u}(t_m)\|_2^2 + \beta\Big(c_r(t_m) - c(t_m)\Big)^2 \\
&+ \int_{t_0}^{t_m} \lambda_m^\top(t)\left(\frac{\mathrm{d}a}{\mathrm{d}t} - Aa - B:(aa^\top) - \frac{\mathrm{d}c_r}{\mathrm{d}t}\Psi^\top\partial_x\Phi(\Psi^\top\Phi)^{-1}a\right)\mathrm{d}t \\
&+ \int_{t_0}^{t_m} \mu_m(t)\left(\frac{\mathrm{d}c_r}{\mathrm{d}t} + \frac{p^\top a + a^\top Q a}{\langle\partial_x\Phi(\Psi^\top\Phi)^{-1}a, \partial_x u_0\rangle}\right)\mathrm{d}t \\
&+ \lambda_m^\top(t_0)\Big(a(t_0) - \Psi^\top\widehat{u}(t_0)\Big)
\end{aligned} \tag{9}$$

  - $\lambda_m(t) \in \mathbb{R}^n, \mu_m(t) \in \mathbb{R}$: Lagrangian multipliers.
  - $\partial_a L_m = 0$, $\partial_{c_r} L_m = 0$ give adjoint equations for $\lambda_m(t)$ and $\mu_m(t)$.

Optimization problem of SR-NiTROM
Unconstrained optimization problem:
$\lambda_m(t) \in \mathbb{R}^n, \mu_m(t) \in \mathbb{R}$: Lagrangian multipliers.
$\partial_a L_m = 0$, $\partial_{c_r} L_m = 0$ give adjoint equations for $\lambda_m(t)$ and $\mu_m(t)$.

Optimization problem of SR-NiTROM

$$
\begin{aligned}
\nabla_\Phi L_m = {} & \left(I - \Psi(\Phi^\top \Psi)^{-1}\Phi^\top\right)\left(2e(t_m)a(t_m)^\top - \partial_x^\top \Psi \int_{t_0}^{t_m} \frac{\mathrm{d}c_r}{\mathrm{d}t}\lambda_m a^\top \mathrm{d}t \right. \\
& \left. - \partial_x^\top(\partial_x u_0)\int_{t_0}^{t_m} \frac{\mu_m(p^\top a + a^\top Q a)}{\langle \partial_x \Phi(\Psi^\top\Phi)^{-1}a, \partial_x u_0\rangle^2} a^\top \mathrm{d}t\right)(\Phi^\top\Psi)^{-1} \qquad (10a) \\
\nabla_\Psi L_m = {} & -2\Phi(\Psi^\top\Phi)^{-1}a(t_m)e(t_m)^\top \Phi(\Psi^\top\Phi)^{-1} \\
& - \int_{t_0}^{t_m} \partial_x \Phi(\Psi^\top\Phi)^{-1} a \lambda_m^\top \frac{\mathrm{d}c_r}{\mathrm{d}t}\mathrm{d}t \\
& + \int_{t_0}^{t_m} \Phi(\Psi^\top\Phi)^{-1} a \lambda_m^\top \frac{\mathrm{d}c_r}{\mathrm{d}t} \Psi^\top \partial_x \Phi(\Psi^\top\Phi)^{-1}\mathrm{d}t \\
& + \int_{t_0}^{t_m} \Phi(\Psi^\top\Phi)^{-1} a \mu_m \frac{p^\top a + a^\top Q a}{\langle \partial_x \Phi(\Psi^\top\Phi)^{-1}a, \partial_x u_0\rangle^2}(\partial_x u_0)^\top \partial_x \Phi(\Psi^\top\Phi)^{-1}\mathrm{d}t \\
& - \hat{u}(t_0)\lambda_m(t_0)^\top \qquad (10b)
\end{aligned}
$$

Optimization problem of SR-NiTROM

Optimization problem of SR-NiTROM

$$\nabla_A L_m = -\int_{t_0}^{t_m} \lambda_m a^\top \mathrm{d}t \tag{11a}$$

$$\nabla_B L_m = -\int_{t_0}^{t_m} \lambda_m \otimes a \otimes a \mathrm{d}t \tag{11b}$$

$$\nabla_p L_m = \int_{t_0}^{t_m} \frac{\mu_m a}{\langle \partial_x \Phi(\Psi^\top \Phi)^{-1} a, \partial_x u_0 \rangle} \mathrm{d}t \tag{11c}$$

$$\nabla_Q L_m = \int_{t_0}^{t_m} \frac{\mu_m a a^\top}{\langle \partial_x \Phi(\Psi^\top \Phi)^{-1} a, \partial_x u_0 \rangle} \mathrm{d}t \tag{11d}$$

$$-\frac{\mathrm{d}\lambda_m}{\mathrm{d}t} = \left(\frac{\partial g}{\partial a}\right)^\top \lambda_m - \left(\frac{\partial h}{\partial a}\right)^\top \mu_m \tag{11e}$$

$$\lambda_m(t_m) = -2(\Phi^\top \Psi)^{-1} \Phi^\top e(t_m) \tag{11f}$$

Optimization problem of SR-NiTROM
(11a) (11b) (11c) (11d) (11e) (11f)

Optimization problem of SR-NiTROM

$$g(a) = Aa + B : (aa^\top) + \frac{\mathrm{d}c_r}{\mathrm{d}t}\Psi^\top \partial_x \Phi(\Psi^\top \Phi)^{-1} a \tag{12a}$$

$$h(a) = \frac{p^\top a + a^\top Q a}{\langle \partial_x \Phi(\Psi^\top \Phi)^{-1} a, \partial_x u_0 \rangle} \tag{12b}$$

$$\mu_m(t) = \lambda_m^\top(t)\Psi^\top \partial_x \Phi(\Psi^\top \Phi)^{-1} a(t) - 2\beta(c_r(t_m) - c(t_m)) \tag{12c}$$

Optimization problem of SR-NiTROM

$$g(a) = Aa + B : (aa^\top) + \frac{\mathrm{d}c_r}{\mathrm{d}t}\Psi^\top \partial_x \Phi(\Psi^\top \Phi)^{-1} a \tag{12a}$$

$$h(a) = \frac{p^\top a + a^\top Q a}{\langle \partial_x \Phi(\Psi^\top \Phi)^{-1} a, \partial_x u_0 \rangle} \tag{12b}$$

$$\mu_m(t) = \lambda_m^\top(t)\Psi^\top \partial_x \Phi(\Psi^\top \Phi)^{-1} a(t) - 2\beta(c_r(t_m) - c(t_m)) \tag{12c}$$